CTRL-REC: CONTROLLING RECOMMENDER SYSTEMS WITH NATURAL LANGUAGE

Micah Carroll, Adeline Foote, Marcus Williams, Anca Dragan, Brad Knox*, Smitha Milli*

ABSTRACT

When users are dissatisfied with recommendations from a recommender system, they often lack fine-grained controls for changing them. Large language models (LLMs) offer a solution by allowing users to guide their recommendations through natural language *requests* (e.g., "I want to see respectful posts with a different perspective than mine"). However, integrating these user requests into traditional recommender systems, which focus on predicting user interaction with specific items, remains a necessary challenge to overcome for practical applications. We propose a method, **CTRL-Rec**, that allows for natural language control of traditional recommender systems in real-time with computational efficiency. Specifically, at training time, we use an LLM to simulate whether users would approve of items based on their language requests, and we train embedding models that approximate such simulated judgments. We then integrate these user-request-based predictions into the standard weighting of signals that traditional recommender systems optimize. At deployment time, we require only a single LLM embedding computation per user request, allowing for real-time control of recommendations. In experiments with the MovieLens dataset, our method consistently allows for fine-grained control across a diversity of requests.

1 INTRODUCTION

When users are unhappy with the recommendations provided by a recommender system, they often find themselves without effective means to alter them. Traditional controls, such as "See less often" or "Not interested" buttons, frequently fail to adjust future recommendations in a way that aligns with users' intents (Mozilla, 2023). Other system-specific controls like genre filters ("Comedy", "Action", etc.) or topic toggles ("Gaming", "Sports", "French Grammar", etc.) limit users to predefined categories. In reality, users often have nuanced, ambiguous requests that cannot be addressed by current mechanisms, e.g., "show me something thought-provoking but not too heavy" or "I want content that helps me learn about other perspectives."

The absence of effective controls is disappointing, as they could act as an important counter-balance to the engagement-focused[1] nature of modern recommender systems (Ekstrand & Willemsen, 2016; Kleinberg et al., 2024a; Morewedge et al., 2023; Lazar et al., 2024). Without user controls, users can only influence their recommendations through changing their engagement patterns over a period of time—a strategy that users do sometimes resort to with limited success (Burrell et al., 2019). However, having effective real-time controls would allow users to immediately adjust content when their preferences differ in a new context (e.g., watching movies with family versus friends). Individuals would also be able to align the recommender system with reflective or aspirational preferences that aren't captured by their engagement history. For instance, a user might want to see thoughtful long-form history videos, even if haven't typically engaged with such content.

Effective user controls may also have benefits at a broader, societal level. While this remains a topic of ongoing debate (Guess et al., 2023), many have theorized that by optimizing for user engagement, current ranking algorithms may replicate existing human biases (Brady et al., 2023; Agan et al., 2023; Morewedge et al., 2023), ultimately leading to increased social misperception and conflict.

[1]Modern recommender systems are primarily based on optimizing user engagement, e.g., likes, clicks, replies, etc. Platforms do also incorporate non-engagement signals like user controls and surveys to some extent (Cunningham et al., 2024), however, current integrations are typically limited by the sparsity of existing non-engagement signals.

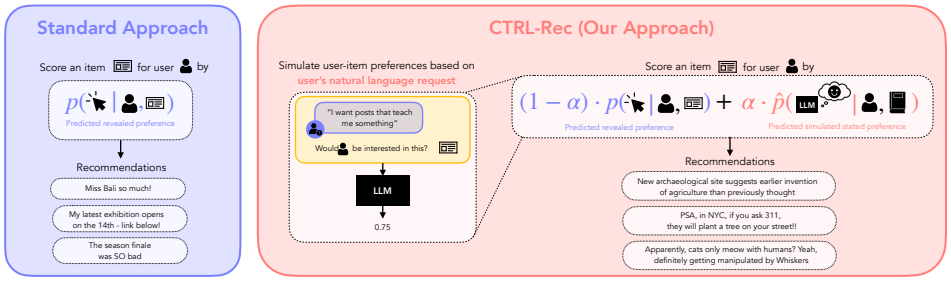

Figure 1: Our approach differs from standard engagement recommenders by allowing to optimize for both revealed preferences (through engagement signals) and stated preferences (through natural language requests). This allows users to explicitly control their recommendations while maintaining the benefits of engagement-based recommendation.

For example, research has found that users tend to engage with inflammatory, divisive content even when they express a desire to see less of it (Milli et al., 2025; Rathje et al., 2024). Therefore, improving control mechanisms that allow users to better align recommender systems with their reflective preferences could be a potential avenue for improving discourse on these platforms.

CTRL-Rec: Controlling Recommender Systems With Natural Language. Natural language can serve as an intuitive and flexible interface for controlling recommender systems. Imagine that a user could simply directly say "I want content that helps me learn about other perspectives," (or any of the other requests in Figure 2) and have their recommendations update immediately in real-time. We contribute a method, **CTRL-Rec**, that allows for this kind of real-time natural language control in a computationally efficient manner.

Our approach is built around two central ideas. First, we overcome the "type mismatch" between conversational and traditional recommender systems. Traditional recommender systems are primarily based upon predicting user-item level interactions, e.g., whether a user will like a particular post. However, natural language requests are broad and free-form and may apply to many items at once, e.g., "I want to see posts that are funny and witty but not mean." To make these free-form requests compatible with traditional recommender systems, we use LLMs to simulate users' judgments of particular items based upon their natural language requests.

Second, building upon ideas from dense retrieval (Izacard & Grave, 2021; Karpukhin et al., 2020; Khattab & Zaharia, 2020), we allow for real-time control by distilling these LLM judgments for computational efficiency. Consider generating recommendations from m candidate items. A naive approach to obtaining simulated user-item level judgments would require m LLM queries each time a user has a new natural language request. Instead, we distill these LLM judgments into efficient embedding models that require only one LLM embedding computation per user request. We then integrate these user-request-based predictions into the standard weighting of signals that traditional recommender systems optimize.

In summary, our contributions are:

1. **A framework for natural language control of recommender systems** that allows users to express their immediate or long-term preferences through free-form and flexible requests.
2. **A scalable method for integrating language-based controls** into traditional recommender systems, allowing for real-time control of recommender systems. Only one LLM embedding computation is required per user request.
3. **Empirical validation on MovieLens** using both genre-specific and subjective, open-ended controls. We find that our approach effectively steers recommendations according to user requests while maintaining engagement quality.

2 CTRL-Rec: Control Through Language for Recommendations

Our system integrates natural language control into traditional recommender systems by introducing a novel preference prediction component that estimates how well items align with user-specified requests.

Figure 2: CTRL-Rec has the potential to be able to handle many kinds of user requests, unlocking many novel forms of user control.

Traditional Recommender Scoring. Modern recommender systems typically optimize a weighted linear combination of different user-item signals (Milli et al., 2023; Cunningham et al., 2024; Smith, 2021; Twitter, 2023). Generally, for a given user u, the score of an item i is computed as

$$\text{score}_{\text{base}}(u, i) = \sum_k w_k f_k(u, i) \tag{1}$$

where each function f_k returns the value of the k-th user-item signal and w_k is the weight on that signal. The different signals $f_k(u, i)$ are typically probabilities or scores for how likely the user u is to interact with item i in different ways, e.g., liking, commenting, etc. Some signals may also be independent of the user u, for example, a signal for the likelihood that the item i violates content moderation standards.

Request-aware Recommender Scoring in CTRL-Rec. Our key insight is that we can naturally incorporate natural language controls into this framework by explicitly incorporating the user's natural language request r as an independent signal. Our updated scoring function is defined as:

$$\text{score}_{\text{CTRL-Rec}}(u, i, r) = \underbrace{\text{score}_{\text{base}}(u, i)}_{\text{revealed preference + other signals}} + w_{\text{control}} \underbrace{v(u, i, r)}_{\text{stated preference}} \tag{2}$$

Here, the function $v(u, i, r)$ represents *how satisfied user u would be with item i given their request r*, and the parameter w_{control} modulates the trade-off between engagement (the user's revealed preferences) and the user's preferences stated in natural language. The nature of the user request r can be very flexible. It could be a simple immediate request or search, or represent more complex, long-term preferences that users would like the recommender system to *always* follow (e.g. "Never recommend me war movies") or only follow *under certain conditions* (e.g. "I do not want to see political content after 8pm").[2]

Platforms typically select the weights on different signals through A/B tests (Cunningham et al., 2024; Twitter, 2023; Milli et al., 2023). By tuning the w_{control} weight, the platform (or potentially even the user)[3] can find an effective balance between the user's revealed and stated preferences. Also note that while we focus on a linear interpolation between engagement and stated preferences (to align with industry practice), other promising combination strategies are possible (Milli et al., 2021), such as threshold-based filtering or multiplicative scoring.

2.1 Value Prediction

The key challenge for our method is estimating $v(u, i, r)$—how well an item aligns with a user's request—and doing so efficiently. A naive approach, which we call **direct LLM scoring**, is to ask an LLM directly to rate how well each candidate item matches the user's request. However, the direct LLM scoring is computationally infeasible for at-scale deployments: it requires performing m LLM queries per user request, where m is the number of items in the candidate pool.[4]

[2]For the latter case, the user state u must contain sufficient information.

[3]Platforms could easily expose an interface that allows the user to select how to trade-off between their language request (stated preferences) and engagement (revealed preferences).

[4]One might wonder if it is possible to simply do one LLM query per user request but ask the LLM to generate ratings for all m items. When m is large, this request requires a long context, particularly if the

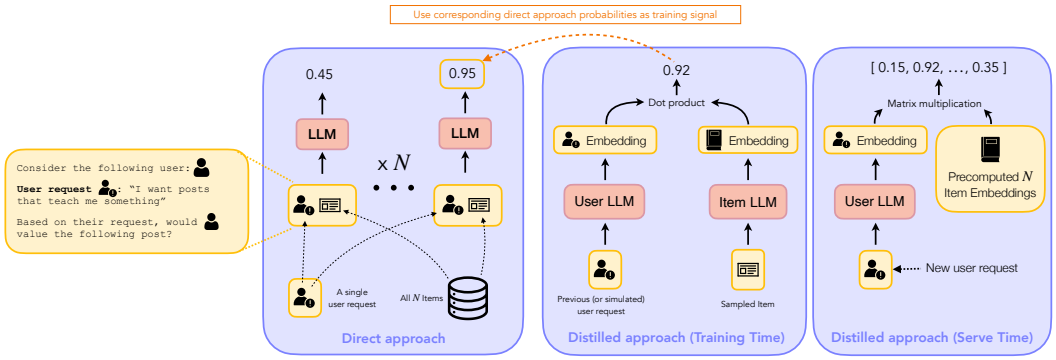

Figure 3: **Overview of our method.** We train a distilled model to approximate LLM judgements of whether a user would like a specific item based on their natural language request. This obviates the need for LLM calls (apart from a single request embedding operation) at test time.

To address this, we can distill the LLM's judgments into a more efficient scoring function—what we call the **distilled approach**—that reduces the computation needed at deployment time from m LLM queries per user request to just one LLM embedding computation. Specifically, we fine-tune LLM embedding models f and g to approximate LLM judgments: $v(u, i, r) \approx f(u, r)^T g(i)$. The item embeddings $g(i)$ can be computed offline, meaning that at deployment time, only one LLM embedding computation per user request (plus a matrix multiplication) is needed. Moreover, as we show in Appendix B.1 this approach can be further optimized by batching simultaneous user requests together. While the idea of using LLM embeddings for engagement prediction has been explored before, we are not aware of prior work on distilling LLMs for this specific item-value prediction task which is conditioned on user's natural language requests.

3 Experiments

In our experiments, we test the ability of our approach to accommodate a range of requests ranging from immediate requests (e.g. specific genres) to more ambiguous and open-ended requests. For simplicity and clarity, the base recommender system predicts one signal: user engagement, in this case, whether the user will watch a movie or not. The score function for our approach then becomes $\text{score}_{\text{CTRL-Rec}}(u, i, r) = w_{\text{eng}} \cdot f_{\text{eng}}(u, i) \; + \; w_{\text{control}} \cdot v(u, i, r)$.

We evaluate our method on the following three tasks:

1. **Genre Requests:** Can CTRL-Rec effectively steer recommendations toward a specific genre that matches the user's past engagement history, and achieve both a good level of steering without compromising much on engagement?
2. **Open-Ended Requests:** How well does CTRL-Rec handle more complex, open-ended user requests?

We evaluate recommendations across different w_{control} values from Equation (2), with the intent of measuring the trade-off between honoring users' explicit requests and maintaining high engagement. To measure *request satisfaction*, we use 'genre accuracy' for genre-specific requests (the fraction of top-k recommendations containing the user's requested genre), and use 'feed-level LLM judgements of request satisfaction' for open-ended requests (using a different LLM than that used for the direct approach).

3.1 Experimental Setup

Dataset. For all our experiments, we use the MovieLens 100k dataset, which contains 100k ratings across 1682 movies and 943 users (Harper & Konstan, 2015). To provide richer item-level infor-

items are also represented by richer natural language descriptions as in Section 3. The long context makes this approach either infeasible or slow.

mation for the LLM simulator, we generate a summary of each movie using GPT-4o-mini.[5] These summaries include plot, themes, tone, and target audience information.

Request-Aware Item-Value Prediction. We explore both direct and distilled approaches to computing $v(u, i, r)$. For the direct approach, we use a Llama-3.1-8B-Instruct model to rate the movie's match with the request, as described in Appendix B. For the distilled approach, we use copies of Stella-en-400M-v5 as our LLM embedding models: we use two separate copies of such a model, and use them respectively to embed users and items. As described in Section 2.1, we fine-tune these two copies such that the dot product of the resulting embeddings approximates LLM judgments from the direct LLM scoring approach. In our experiments, we evaluate both the distilled approach and the direct approach, with the direct approach functioning as a gold standard. As a baseline, we also consider simply using the dot product between the untrained user and item embeddings to rank items, as done in prior work (Karpukhin et al., 2020).

Engagement Recommender. As our engagement recommender that we interpolate with, we use a BiVAE (Bi-Variational Autoencoder) model trained on historical user-movie interactions. The model was trained using the Microsoft recommenders module (Graham et al., 2019) with their recommended hyperparameters for MovieLens recommendation tasks (and was chosen as it was the best performing model in their suite as reported by them). We split each user's data chronologically, using the earliest 50% of ratings for training and the remaining 50% for testing.

Score Aggregation. An implementation detail we found to be crucial for effective performance is how we combine engagement scores with request-based item-value predictions. Rather than directly normalizing and combining the raw scores, we first convert both engagement scores and request-based scores into ranks, then normalize these ranks to [0,1] before combining them according to Equation (2). This rank-based approach proved qualitatively more effective than direct score normalization for balancing engagement and request satisfaction (as discussed in Section 3.3).

User Request Generation. In lieu of real user natural language requests, we simulate personalized user requests for the MovieLens user's using Llama-3.3-70B-Instruct. For the genre requests, we instruct the LLM to generate a request which mentions a specific genre which seems consistent with their prior engagements ("I'd like to watch a good comedy"). For the open-ended requests, we instruct the LLM to generate requests which are more complex and which may be more ambiguous, also consistent with the user's previous engagements (e.g. "I want to watch a heartwarming movie that's thought-provoking").

The full prompting details for all LLM components are provided in Appendix A.

Quality of the Distilled Approach. We also benchmarked the quality of the predictions made by the distilled approach. In Figure 6, we show the distribution of LLM preference scores for the distilled approach and the direct LLM scoring approach on the training distribution. We see that the distilled approach is able to match the quality of the direct LLM scoring approach relatively well, with a Spearman rank correlation of 0.77.

3.2 RESULTS

Genre-Specific Requests. Table 1 and Figure 4 (left) show that the distilled approach achieves similar performance to direct LLM scoring in terms of request satisfaction, despite being many orders of magnitude more computationally efficient. This is particularly noteworthy given that we use a relatively small embedding model (Stella-400M) compared to the LLM used for the direct approach (Llama-3.1-8B). For genre-specific requests, we see that also a simple baseline of untrained LLM embeddings does quite well, returning the correct genre for ∼90% of recommended items.

Somewhat surprisingly, we find that the direct LLM scoring approach produces qualitatively reasonable recommendations even when setting $w_{\text{control}} = 1$ (i.e., completely ignoring engagement signals). This is noteworthy because naively finding items that maximize these LLM-value judgements would be expected to amplify noise and lead to poor recommendations - if the LLM's confidence scores were poorly calibrated or noisy, selecting items that maximize these scores should overfit to this noise. The fact that this doesn't happen in practice suggests that modern LLMs are surprisingly well-calibrated for this type of preference judgment task.

[5] All models were implemented and run on through UC Berkeley systems.

Related to the above, we see that especially for the distilled and untrained methods, request satisfaction slightly degrades for very high w_{control} values. Ultimately, we think that in the case of the distilled model, this drop is due to the phenomenon hypothesized above, and the increase in noise in estimates introduced by the distillation process itself. This doesn't affect performance as much for lower w_{control} values because the top recommended items will be "regularized by the engagement ranking", which ensures tie-breaking between movies that seem similarly related to the query are resolved in favor of more popular movies. Indeed, we find that a lot of the drop in the accuracy (at least for the distilled approach) is driven by the recommended movies being very niche, and having incorrect listing of genres in the MovieLens 100k dataset (that have since been updated on IMDB). Interestingly, the direct approach is less susceptible to this effect, despite the incorrect genre labels in the MovieLens 100k dataset: we hypothesize that this is because the LLM judgements have a slight implicit popularity bias (serving a similar regularizing role), which is lost during distillation.

Another interesting phenomenon (that also occurs for the open-ended requests) is that the distilled model does somewhat better than the direct approach for lower values of w_{control}. We hypothesize this is mostly an artifact of how errors are distributed during distillation (shown in Figure 6).

Open-Ended Requests. For more complex, open-ended requests (e.g., "I want something thought-provoking but not too heavy"), as seen in Table 1 and Figure 4 (right), the gap between methods increases. The distilled approach again tracks the direct LLM approach closely, though with a slightly larger gap than in the genre-specific case. This suggests that while our embedding-based approximation works well for concrete genre requests, there may be room for improvement in capturing more nuanced preferences. Future work could explore whether this gap can be closed by using larger embedding models or more sophisticated distillation techniques, and for requests that one would expect to be especially challenging for untrained embeddings (e.g. "movies like those of Miyazaki but not by Miyazaki").

Note that we still observe decreases in request satisfaction for high values of w_{control} (for both the distilled and untrained embedding approaches). In this case, qualitatively we do find that the feed quality is reduced for higher values and not due to issues with the accuracy metric as in the case of the genre experiment. Indeed, "feed quality"—which is ultimately what we want—should likely take into account item popularity (and the LLM judge implicitly takes that into account when scoring feeds). The direct approach is able to do this subtly, but this capacity is lost in the distillation (and was never present for the untrained embedding approach). In light of that, we hypothesize that in this case, the drop is entirely due to the lack of the regularization effect of the engagement recommender.

Trade-offs between stated preference satisfaction and engagement. We report the Pareto frontiers for the three methods we consider for our two experimental setups in Section 3. However, as discussed in that section, the trade-offs we measure are likely highly conservative, as one would expect users to engage with results that match their requests (and static datasets for measuring engagement metrics are not able to capture that). Despite their conservativeness, our results suggest that at a minimum, very large increases in request satisfaction are achievable with very little engagement cost.

3.3 Qualitative Examples

As discussed above, the true performance of our method is better assessed through qualitative examples (shown below) and direct user interaction. To facilitate this, we have made an interactive demo available here.

Table 1: **Maximum Request Satisfaction Achieved by Different Methods.** As one would expect, the direct approach performs best overall, followed by the distilled approach and then the untrained embeddings. The gap between trained and untrained embeddings is notably larger for open-ended requests, suggesting that distilling LLM judgements is especially important for handling more complex, nuanced requests.

	Genre Requests			**Open-ended Requests**		
Method	Max Satisfaction	w_{control}	% vs Untrained	Max Satisfaction	w_{control}	% vs Untrained
Direct LLM	0.952 ± 0.004	0.97	+5.9%	0.919	1.00	+17.0%
Distilled	0.954 ± 0.004	0.97	+6.1%	0.868	0.95	+10.6%
Untrained Embeddings	0.899 ± 0.005	0.90	–	0.785	0.95	–

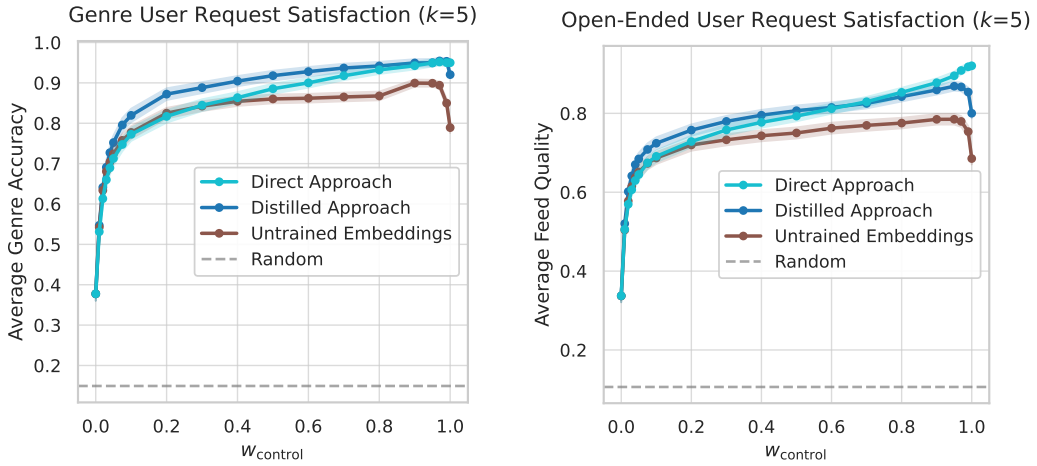

Figure 4: **Request Satisfaction as we increase the weight of w_{control}.** As one would expect, we see that as we increase w_{control}, metrics of request satisfaction generally increase across all methods. Importantly, we see that the request satisfaction for the direct approach (when picking the best performing w_{control}) is highest, followed closely by the distilled approach. We see that the untrained embeddings also perform relatively well for these two tasks—with the gap being bigger for the open-ended generation task, as one would expect.

Table 2 provide concrete examples of how recommendations change as we vary w_{control} for a user requesting ““Can you give me a good movie for a fun night with my 8 year old kids?” (a request that stumps Netflix's currently deployed search system). At $w_{\text{control}} = 0$ (pure engagement), the system recommends highly-rated but potentially inappropriate films like Pulp Fiction. As w_{control} increases, recommendations shift toward children's content while maintaining reasonable engagement scores. At $w_{\text{control}} = 1$, recommendations consist entirely of family-friendly films.

Qualitatively, we find that using direct LLM scoring instead of our distilled approach leads to somewhat better behavior (Table 4). We find it also easiest to see how alternate scoring aggregation schemes fail through these qualitative examples: Table 3 demonstrates that naive linear score weighting leads to less balanced transitions between engagement and request satisfaction compared to our rank-based approach, suggesting the importance of score normalization via ranking.

Table 2: Impact of w_{control} on recommendation rankings using a distilled approach when user requests “Can you give me a good movie for a fun night with my 8 year old kids?”.

Title	Genres	Request Rank	Eng. Rank	Combined Rank
$w_{control} = 0$: *Pure Engagement-Based (No Request Consideration)*				
Pulp Fiction	Crime, Drama	1004	1	1
Army of Darkness	Action, Adventure, Comedy, Horror, Sci-Fi	522	2	2
GoldenEye	Action, Adventure, Thriller	639	3	3
The Fifth Element	Action, Sci-Fi	216	4	4
Desperado	Action, Romance, Thriller	905	5	5
$w_{control} = 0.5$: *Equal weighting*				
Toy Story	Animation, Children, Comedy	66	41	54
Liar Liar	Comedy	96	32	64
Jumanji	Action, Adventure, Children, Fantasy, Sci-Fi	1	134	68
Mrs. Doubtfire	Comedy	66	70	68
George of the Jungle	Children, Comedy	56	82	69
$w_{control} = 1$: *Pure Request-Based (No Engagement Consideration)*				
Jumanji	Action, Adventure, Children, Fantasy, Sci-Fi	1	134	1
A Kid in King Arthur's Court	Adventure, Children, Comedy, Fantasy, Romance, Sci-Fi	3	664	3
First Kid	Children, Comedy	3	574	3
Star Kid	Adventure, Children, Fantasy, Sci-Fi	4	959	4
Rocket Man	Comedy	5	156	5

4 Related Work

User Control of Recommender Systems. Prior research has aimed at better aligning recommender systems with users' stated preferences through the use of structured controls such as "See less often" buttons or genre toggles (Milli et al., 2021; Kleinberg et al., 2024b; Agarwal et al., 2024; Yang et al., 2019). We prioritize natural language control because it allows for the expression of flexible, open-ended, and subjective requests. Research has consistently shown that existing user controls have low usage among users (Cunningham et al., 2024), likely because these controls are often ineffective at shaping recommendations to users' intents (Mozilla, 2023). In contrast, our method is more analogous to widely-used search queries, and allows for natural language control in *real-time*, which we hope aids in the adoption by providing clear and immediate feedback to the user, though further research is needed to validate this hypothesis.

LLMs for Recommender Systems and Retrieval. There has been an increasing amount of research focused on leveraging large language models for recommendation (Wu et al., 2024; Wang et al., 2024; Lin et al., 2024; Huang et al., 2024) and information retrieval more broadly (Karpukhin et al., 2020; Khattab & Zaharia, 2020). Our work sits somewhere between recommendation and information retrieval, as we provide a way for users to provide natural language descriptions of their preferences which are not necessarily one-time searches (e.g. "never show me political content after 10pm"). To integrate these natural language requests into traditional recommender systems, we utilize an LLM to translate these high-level requests into item-level judgments and then distill these simulated LLM judgments for computationally efficient predictions at inference time.

Prior research has directly used LLMs to score the relevance of items to users based on natural language preference strings (Sanner et al., 2023). However, due to the high computational cost associated with LLMs, they are limited to scoring or re-ranking only a small candidate set of items. In contrast, our approach enables real-time control even at the retrieval stage by distilling synthetic LLM judgments. The general idea of generating synthetic labels via an LLM and then distilling has been applied before in dense retrieval (Izacard & Grave, 2021; Bonifacio et al., 2022; Huang & Chen, 2024), although with different types of judgments and for different purposes than our goal here (e.g. scoring "relevance" of answers to user questions). Note that, while in this work we distill the LLM judgments into a dual-encoder architecture (Karpukhin et al., 2020; Bromley et al., 1993), our general framework is agnostic to the exact distillation method used and could leverage other methods from dense retrieval.

The novel goal of our work is to give users explicit control over standard recommendation interfaces through natural language requests. In this way, our work contrasts from other research on providing recommendations limited to a chat interface (Gao et al., 2023; Friedman et al., 2023; He et al., 2023) that do not affect traditional recommender systems. It also contrasts from work that uses natural language features or profiles for recommendation (Mysore et al., 2023b; Kim et al., 2024; Paischer et al., 2024), but where these profiles are extracted or learned, rather than provided by the user as a means of control. Extracted natural language profiles are also common in more recent works on scrutable recommender systems (Radlinski et al., 2022; Penaloza et al., 2025; Ramos et al., 2024; Gao et al., 2025; Mysore et al., 2023a), which also allow for user control via editing of said profiles. However, CTRL-Rec differs both in methodology for connecting natural language to recommendations, and extends the paradigm to accommodate immediate requests (which are non persistent), enabling more dynamic control.

5 Discussion

Our results demonstrate that natural language requests can be effectively integrated into traditional recommender systems, providing users with fine-grained control while maintaining engagement quality. The success of our distilled approach suggests that LLM-based preference simulation can be made computationally efficient enough for practical deployment, requiring only a single embedding computation per user request.

Limitations. Our work has several important limitations. First, while our qualitative examples suggest the method behaves reasonably, we lack real-world user studies to validate that our approach

actually improves user satisfaction in practice. Deployment studies would be valuable for understanding how users interact with natural language controls and whether they find them helpful for achieving their goals. The artificial nature of our simulated user requests is also a limitation - while we attempt to generate realistic requests, they may not fully capture the diversity and complexity of how users would naturally express their preferences.

Second, our evaluation is limited to a single dataset (MovieLens) and domain (movies). While this allows us to thoroughly analyze our method's behavior, it leaves open questions about generalization to other recommendation contexts like social media, e-commerce, or news. Each domain may present unique challenges in terms of both the nature of user requests and the characteristics of items being recommended.

This limitation connects to a deeper question about our method's reliance on LLMs: the MovieLens dataset consists primarily of well-known movies that LLMs have likely encountered during training. While we provide LLMs with rich textual descriptions of items, our experiments don't directly test whether these descriptions alone would be sufficient for accurate preference simulation on completely novel items. We hypothesize that given sufficiently detailed item descriptions, LLMs should be able to make reasonable preference judgments even for unseen items – much as humans can judge whether a movie matches their preferences based on its description, without having seen it. However, validating this hypothesis would require evaluation on datasets containing novel items with rich descriptions. Alternatively, our framework would also naturally extend to multimodal models.

Conclusion. We have presented CTRL-Rec, a framework for incorporating natural language user controls into traditional recommender systems. Our key insight is that LLMs can be used to simulate how well items align with user requests, and these simulated judgments can be distilled and decomposed into efficient dot products between user-request and item embeddings. This decomposition allows us to reduce the computational cost by orders of magnitude, requiring only one LLM embedding computation per user request rather than per item. Our experiments demonstrate that this approach allows for effective steering of recommendations according to user requests while maintaining high engagement metrics. The method is computationally efficient and practical for real-world deployment. We believe this work represents a step toward recommender systems that better balance engagement optimization with explicit user preferences and control.

REFERENCES

Amanda Y Agan, Diag Davenport, Jens Ludwig, and Sendhil Mullainathan. Automating Automaticity: How the Context of Human Choice Affects the Extent of Algorithmic Bias. Technical report, National Bureau of Economic Research, 2023.

Arpit Agarwal, Nicolas Usunier, Alessandro Lazaric, and Maximilian Nickel. System-2 recommenders: Disentangling utility and engagement in recommendation systems via temporal point-processes. In *The 2024 ACM Conference on Fairness, Accountability, and Transparency*, pp. 1763–1773, 2024.

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. InPars: Data Augmentation for Information Retrieval using Large Language Models, February 2022. URL http://arxiv.org/abs/2202.05144. arXiv:2202.05144 [cs].

William J Brady, Joshua Conrad Jackson, Björn Lindström, and MJ Crockett. Algorithm-mediated social learning in online social networks. *Trends in Cognitive Sciences*, 2023.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "Siamese" time delay neural network. In *Proceedings of the 7th International Conference on Neural Information Processing Systems*, NIPS'93, pp. 737–744, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

Jenna Burrell, Zoe Kahn, Anne Jonas, and Daniel Griffin. When users control the algorithms: Values expressed in practices on twitter. *Proc. ACM Hum.-Comput. Interact.*, 3(CSCW), November 2019. doi: 10.1145/3359240. URL https://doi.org/10.1145/3359240.

Tom Cunningham, Sana Pandey, Leif Sigerson, Jonathan Stray, Jeff Allen, Bonnie Barrilleaux, Ravi Iyer, Smitha Milli, Mohit Kothari, and Behnam Rezaei. What we know about using non-engagement signals in content ranking. *arXiv preprint arXiv:2402.06831*, 2024.

Michael D. Ekstrand and Martijn C. Willemsen. Behaviorism is not enough: Better recommendations through listening to users. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, pp. 221–224, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450340359. doi: 10.1145/2959100.2959179. URL https://doi.org/10.1145/2959100.2959179.

Luke Friedman, Sameer Ahuja, David Allen, Zhenning Tan, Hakim Sidahmed, Changbo Long, Jun Xie, Gabriel Schubiner, Ajay Patel, Harsh Lara, Brian Chu, Zexi Chen, and Manoj Tiwari. Leveraging Large Language Models in Conversational Recommender Systems, May 2023. URL http://arxiv.org/abs/2305.07961. arXiv:2305.07961 [cs].

Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. Chat-REC: Towards Interactive and Explainable LLMs-Augmented Recommender System, April 2023. URL http://arxiv.org/abs/2303.14524. arXiv:2303.14524 [cs].

Zhaolin Gao, Joyce Zhou, Yijia Dai, and Thorsten Joachims. End-to-end Training for Recommendation with Language-based User Profiles, February 2025. URL http://arxiv.org/abs/2410.18870. arXiv:2410.18870 [cs].

Scott Graham, Jun-Ki Min, and Tao Wu. Microsoft recommenders: tools to accelerate developing recommender systems. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, pp. 542–543, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362436. doi: 10.1145/3298689.3346967. URL https://doi.org/10.1145/3298689.3346967.

Andrew M. Guess, Neil Malhotra, Jennifer Pan, Pablo Barberá, Hunt Allcott, Taylor Brown, Adriana Crespo-Tenorio, Drew Dimmery, Deen Freelon, Matthew Gentzkow, Sandra González-Bailón, Edward Kennedy, Young Mie Kim, David Lazer, Devra Moehler, Brendan Nyhan, Carlos Velasco Rivera, Jaime Settle, Daniel Robert Thomas, Emily Thorson, Rebekah Tromble, Arjun Wilkins, Magdalena Wojcieszak, Beixian Xiong, Chad Kiewiet de Jonge, Annie Franco, Winter Mason, Natalie Jomini Stroud, and Joshua A. Tucker. How do social media feed algorithms affect attitudes and behavior in an election campaign? *Science*, 381(6656):398–404, 2023.

doi: 10.1126/science.abp9364. URL https://www.science.org/doi/abs/10.1126/science.abp9364.

F. Maxwell Harper and Joseph A. Konstan. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19, December 2015. ISSN 2160-6455. doi: 10.1145/2827872. URL https://doi.org/10.1145/2827872.

Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley. Large Language Models as Zero-Shot Conversational Recommenders. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 720–730, October 2023. doi: 10.1145/3583780.3614949. URL http://arxiv.org/abs/2308.10053. arXiv:2308.10053 [cs].

Chao-Wei Huang and Yun-Nung Chen. PairDistill: Pairwise Relevance Distillation for Dense Retrieval. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 18225–18237, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1013. URL https://aclanthology.org/2024.emnlp-main.1013/.

Chengkai Huang, Tong Yu, Kaige Xie, Shuai Zhang, Lina Yao, and Julian McAuley. Foundation Models for Recommender Systems: A Survey and New Perspectives, February 2024. URL http://arxiv.org/abs/2402.11143. arXiv:2402.11143 [cs].

Gautier Izacard and Edouard Grave. Distilling knowledge from reader to retriever for question answering. In *ICLR*, 2021.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense Passage Retrieval for Open-Domain Question Answering, September 2020. URL http://arxiv.org/abs/2004.04906. arXiv:2004.04906 [cs].

Omar Khattab and Matei Zaharia. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, pp. 39–48, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450380164. doi: 10.1145/3397271.3401075. URL https://doi.org/10.1145/3397271.3401075.

Jieyong Kim, Hyunseo Kim, Hyunjin Cho, SeongKu Kang, Buru Chang, Jinyoung Yeo, and Dongha Lee. Review-driven personalized preference reasoning with large language models for recommendation, 2024. URL https://arxiv.org/abs/2408.06276.

Jon Kleinberg, Jens Ludwig, Sendhil Mullainathan, and Manish Raghavan. The inversion problem: Why algorithms should infer mental state and not just predict behavior. *Perspectives on Psychological Science*, 19(5):827–838, 2024a.

Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. The challenge of understanding what users want: Inconsistent preferences and engagement optimization. *Management science*, 70(9): 6336–6355, 2024b.

Seth Lazar, Luke Thorburn, Tian Jin, and Luca Belli. The Moral Case for Using Language Model Agents for Recommendation, October 2024. URL http://arxiv.org/abs/2410.12123. arXiv:2410.12123 [cs].

Jianghao Lin, Xinyi Dai, Yunjia Xi, Weiwen Liu, Bo Chen, Hao Zhang, Yong Liu, Chuhan Wu, Xiangyang Li, Chenxu Zhu, Huifeng Guo, Yong Yu, Ruiming Tang, and Weinan Zhang. How Can Recommender Systems Benefit from Large Language Models: A Survey, July 2024. URL http://arxiv.org/abs/2306.05817. arXiv:2306.05817 [cs].

Smitha Milli, Luca Belli, and Moritz Hardt. From Optimizing Engagement to Measuring Value. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 714–722, 2021.

Smitha Milli, Emma Pierson, and Nikhil Garg. Choosing the right weights: Balancing value, strategy, and noise in recommender systems. *arXiv preprint arXiv:2305.17428*, 2023.

Smitha Milli, Micah Carroll, Yike Wang, Sashrika Pandey, Sebastian Zhao, and Anca D Dragan. Engagement, user satisfaction, and the amplification of divisive content on social media. *PNAS Nexus*, 4(3):pgaf062, 03 2025. ISSN 2752-6542. doi: 10.1093/pnasnexus/pgaf062. URL https://doi.org/10.1093/pnasnexus/pgaf062.

Carey K Morewedge, Sendhil Mullainathan, Haaya F Naushan, Cass R Sunstein, Jon Kleinberg, Manish Raghavan, and Jens O Ludwig. Human bias in algorithm design. *Nature Human Behaviour*, 7(11):1822–1824, 2023.

Mozilla. Youtube regrets: A crowdsourced investigation into youtube's recommendation algorithm. https://foundation.mozilla.org/en/youtube/findings/, 2023. Accessed: 2023-10-17.

Sheshera Mysore, Mahmood Jasim, Andrew McCallum, and Hamed Zamani. Editable User Profiles for Controllable Text Recommendation, October 2023a. URL http://arxiv.org/abs/2304.04250. arXiv:2304.04250 [cs].

Sheshera Mysore, Andrew Mccallum, and Hamed Zamani. Large language model augmented narrative driven recommendations. In *Proceedings of the 17th ACM Conference on Recommender Systems*, RecSys '23, pp. 777–783, New York, NY, USA, 2023b. Association for Computing Machinery. ISBN 9798400702419. doi: 10.1145/3604915.3608829. URL https://doi.org/10.1145/3604915.3608829.

Fabian Paischer, Liu Yang, Linfeng Liu, Shuai Shao, Kaveh Hassani, Jiacheng Li, Ricky Chen, Zhang Gabriel Li, Xialo Gao, Wei Shao, Xue Feng, Nima Noorshams, Sem Park, Bo Long, and Hamid Eghbalzadeh. Preference Discerning with LLM-Enhanced Generative Retrieval, December 2024. URL http://arxiv.org/abs/2412.08604. arXiv:2412.08604 [cs].

Emiliano Penaloza, Olivier Gouvert, Haolun Wu, and Laurent Charlin. TEARS: Textual Representations for Scrutable Recommendations, March 2025. URL http://arxiv.org/abs/2410.19302. arXiv:2410.19302 [cs].

Filip Radlinski, Krisztian Balog, Fernando Diaz, Lucas Dixon, and Ben Wedin. On Natural Language User Profiles for Transparent and Scrutable Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2863–2874, July 2022. doi: 10.1145/3477495.3531873. URL http://arxiv.org/abs/2205.09403. arXiv:2205.09403 [cs].

Jerome Ramos, Hossen A. Rahmani, Xi Wang, Xiao Fu, and Aldo Lipani. Transparent and Scrutable Recommendations Using Natural Language User Profiles, July 2024. URL http://arxiv.org/abs/2402.05810. arXiv:2402.05810 [cs].

Steve Rathje, Claire Robertson, William J Brady, and Jay J Van Bavel. People think that social media platforms do (but should not) amplify divisive content. *Perspectives on Psychological Science*, 19(5):781–795, 2024.

Scott Sanner, Krisztian Balog, Filip Radlinski, Ben Wedin, and Lucas Dixon. Large Language Models are Competitive Near Cold-start Recommenders for Language- and Item-based Preferences, July 2023. URL http://arxiv.org/abs/2307.14225. arXiv:2307.14225 [cs].

Ben Smith. How TikTok reads your mind, Dec 2021. URL https://www.nytimes.com/2021/12/05/business/media/tiktok-algorithm.html.

Twitter. Twitter's Recommendation Algorithm - Heavy Ranker and TwHIN embeddings, Mar 2023. URL https://github.com/twitter/the-algorithm-ml/tree/main/projects/home/recap.

Qi Wang, Jindong Li, Shiqi Wang, Qianli Xing, Runliang Niu, He Kong, Rui Li, Guodong Long, Yi Chang, and Chengqi Zhang. Towards Next-Generation LLM-based Recommender Systems: A Survey and Beyond, October 2024. URL http://arxiv.org/abs/2410.19744. arXiv:2410.19744 [cs].

Marcus Williams, Micah Carroll, Adhyyan Narang, Constantin Weisser, Brendan Murphy, and Anca Dragan. On Targeted Manipulation and Deception when Optimizing LLMs for User Feedback, November 2024. URL http://arxiv.org/abs/2411.02306. arXiv:2411.02306 [cs].

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, Hui Xiong, and Enhong Chen. A Survey on Large Language Models for Recommendation, June 2024. URL http://arxiv.org/abs/2305.19860. arXiv:2305.19860 [cs].

Longqi Yang, Michael Sobolev, Yu Wang, Jenny Chen, Drew Dunne, Christina Tsangouri, Nicola Dell, Mor Naaman, and Deborah Estrin. How intention informed recommendations modulate choices: A field study of spoken word content. In *The World Wide Web Conference*, WWW '19, pp. 2169–2180, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366748. doi: 10.1145/3308558.3313540. URL https://doi.org/10.1145/3308558.3313540.

APPENDIX

A PROMPTS

A.1 MOVIE SUMMARY GENERATION

To ensure consistent and informative movie descriptions, we pre-generate summaries for all movies using the following prompt:

System: You are a knowledgeable film critic. Provide accurate movie summaries.

User: You are tasked with generating a summary of a specific movie. This summary should be maximally helpful for someone deciding whether to recommend the movie to another person based on their preferences. Follow these instructions carefully:

You will be given the following information:

<movie_title>
{movie_title}
</movie_title>

Using this information, create a comprehensive summary of the movie. Your summary should focus on aspects that would be most relevant when considering whether to recommend the movie to someone. Include the following elements:

1. Basic information: Briefly mention the director, main cast, and genre
2. Plot overview: Provide a high-level summary of the plot including spoilers
3. Themes and tone: Describe the main themes explored in the movie and its overall tone (e.g., dark, lighthearted, thought-provoking)
4. Cinematic elements: Highlight notable aspects of cinematography, music, or special effects if they are particularly significant
5. Critical reception: Mention how the movie was generally received by critics and audiences
6. Potential appeal: Describe the types of viewers who might enjoy this movie (e.g., fans of certain genres, people interested in specific themes)
7. Content advisories: Mention any content that might make the movie unsuitable for certain audiences (e.g., violence, sexual content, complex themes)

Keep your summary focused and relevant. Avoid unnecessary details or trivia that wouldn't help in deciding whether to recommend the movie.

A.2 GENRE USER REQUEST GENERATION

For generating user requests focused on specific genres, we use the following prompt:

System: You are an AI generating natural language commands that users might give to a movie recommendation system. You should generate a brief first-person statement about which single movie genre the user wants to see right now. Only mention one genre they want to see (not ones to avoid) and only use genres from this specific list: Drama, Comedy, Romance, Thriller, Action, Crime, Adventure, Children, Mystery, Sci-Fi, Fantasy, Horror. The genre requested should reflect what you think this user would genuinely want to watch next based on their rating history.

User: Consider the following movie ratings history for the user:

<movie_ratings_history>
{movie_ratings_str}
</movie_ratings_history>

Guidelines for the statement:

- Write in first-person perspective
- Do not mention specific movies or ratings
- Only mention one genre you want to see (not ones you want to avoid)
- Only use genres from the provided list
- The genre you request should be representative of what you would likely want to watch next, based on your rating history
- Aim for a natural, conversational tone
- Should be a single sentence mentioning exactly one genre
- Do not mention any other movie aspects besides this specific genre

A.3 Open-Ended User Request Generation

For generating more complex, open-ended user requests, we use the following prompt:

System: You are an AI generating diverse natural language commands that users might give to a movie recommendation system. You should generate a brief first-person statement about which kinds of movies the user wants to see right now.

User: Guidelines for the statement:

- Write in first-person perspective
- Do not mention specific movies or ratings
- Focus on preferences, likes, and dislikes related to movies
- The user may express interest in movies based on any aspect, such as genres, themes, storytelling styles, visual elements, acting, and production quality
- Aim for a natural, conversational, and informal tone, as if you were quickly expressing your preferences. Should almost always be a single sentence, potentially even a very short one
- Vary the structure and focus of the statement to add diversity

Consider the following movie ratings history for the user:

<movie_ratings_history>
{movie_ratings_str}
</movie_ratings_history>

Provide the final statement within <statement> tags. Here is an example output structure (do not copy the content, only the format):

<statement>
Final first-person statement about what kind of movie the user is currently looking for
</statement>

{previous_requests}

Be creative with the style of requests. Some examples of different styles:

- Direct and simple ("I want something funny")
- Descriptive ("Looking for an emotional drama that will make me think")
- Mood-based ("I'm in the mood for something thrilling and suspenseful")
- Preference-focused ("I prefer movies with complex characters and deep themes")
- Contextual ("Need a light comedy for a relaxing evening")
- Comparative ("Want something like my favorite action movies but with more humor")

MAKE SURE you open and close your statement tags correctly, <statement> and </statement>.

A.4 DISTILLED MODEL USER REQUEST GENERATION

For generating diverse training data for the distilled model, we use the following prompt:

System: You are an AI generating diverse natural language commands that users might give to a movie recommendation system. The statements can include mixes of genres, or say they want to avoid certain genres, or both.

User: Here are some previous requests you've generated. Try to keep them diverse and focus on areas you think you may have missed.

<jsonl>
{requests_str}
</jsonl>

Please respond only with a JSONL list of 20 new request strings in a similar format, continuing the id counter from the previous requests. Do not output more than 20 new requests (stop at id 40). MAKE SURE to format the JSONL correctly. Write your response in <jsonl></jsonl> tags.

A.5 FEED-LEVEL LLM JUDGE PROMPT

For evaluating how well a set of recommendations matches a user's request, we use the following prompt:

System: You are an expert movie recommendation system evaluator. Your task is to rate how well a list of recommended movies matches a user's request.

User: Consider a user with the following movie request:

<user_preferences>
{user_preference_text}
</user_preferences>

Here are the top {top_k} movie recommendations for this user:
{recommendations_str}

Rate how well these recommendations match the user's request on a scale of 1-5:

- 1 = Poor match, recommendations don't reflect user request
- 3 = Good match, many recommendations align with request
- 5 = Excellent match, all recommendations strongly align with request

Answer only with an integer from 1 to 5. Do not include any other text.

Answer:

B EXTRACTING SCORES FROM LLM

To obtain more calibrated granular scores from LLM judgements, we follow the methodology of Williams et al. (2024). Rather than directly requesting a single rating on a scale of 1-5, we leverage the model's underlying probability distribution. Specifically, we extract the logprobs for tokens "1" through "5", normalize them into a probability distribution $P(r)$, and compute the expected rating as $\sum_{r=1}^{5} r \cdot P(r)$. To verify model comprehension, we ensure the total probability mass on these five tokens exceeds 0.9.

B.1 COMPUTATIONAL EFFICIENCY AND QUALITY OF DISTILLATION

Before considering the results of the experiments described in Section 3, we discuss the efficiency and quality of our trained distilled model.

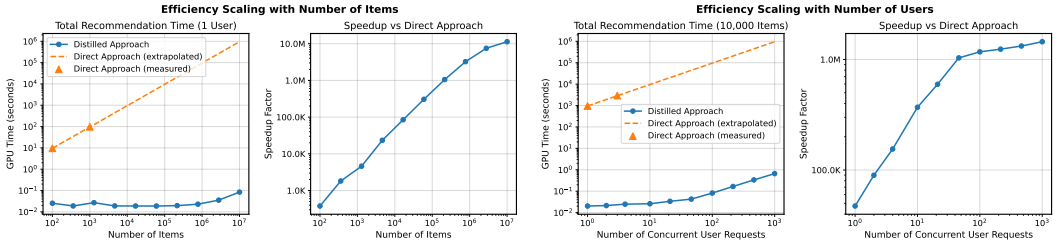

Figure 5: **Computational efficiency: the distilled approach fares far better than the direct approach at scale.** The distilled approach requires many orders of magnitude less computational resources than the direct approach for generating recommendations. Moreover, the speed-up factor for the distilled approach grows both with the number of candidate items and concurrent user requests. With 10^7 items and 1000 concurrent user requests, *we estimate that the distilled approach would likely be at least* 10^8 *times more efficient than direct LLM scoring*. All evaluations were done on a single NVIDIA RTX A6000 GPU.

Distilled Model Computational Efficiency. We benchmarked the computational efficiency of the distilled approach compared to the direct LLM scoring approach. As seen in Figure 5, the computational resources required for the direct approach grow linearly with the number of candidate items and the number of concurrent user requests. Given the high starting cost even for small item sets and a single user, it is clearly infeasible to deploy at scale without massive computational resources and parallelization. While we expect the distilled approach will also grow linearly at large scales, it's far lower starting cost makes it much more applicable to real-world recommendation scenarios.

C Additional Experimental Results

C.1 Distillation Quality

In Figure 6 we show the distribution of predictions vs their respective target labels.

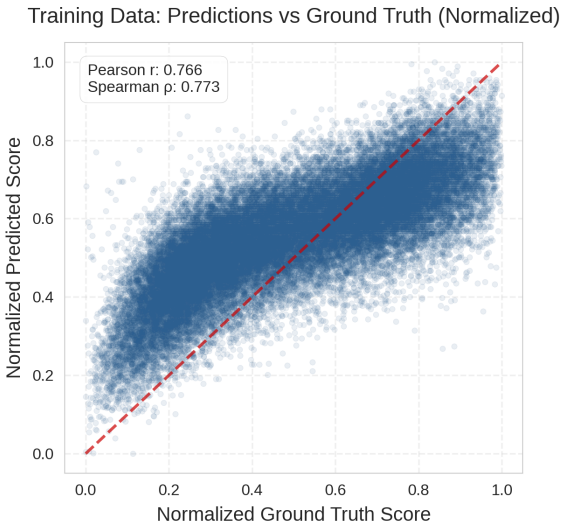

Figure 6: **Distilled approach prediction quality.**

C.2 Trade-offs between engagement and request satisfaction

To measure *engagement*, we use standard metrics: Precision@k (fraction of recommended items in the user's test set), Recall@k (fraction of test set items in the recommendations), and NDCG@k (Normalized Discounted Cumulative Gain at k).

Figure 7 shows the trade-off between engagement metrics and genre accuracy as we vary w_{control} in Equation (2). The results demonstrate that our approach can substantially increase genre accuracy with minimal impact on engagement metrics. For example, using the direct LLM approach, we can

increase genre accuracy from $\sim 38\%$ to $> 90\%$ while only reducing precision by $< 5\%$ percentage points. This suggests that many high-engagement items match users' requested genres, and our method (with appropriate w_{control}) is able to effectively identify and surfacethese items when they exist.

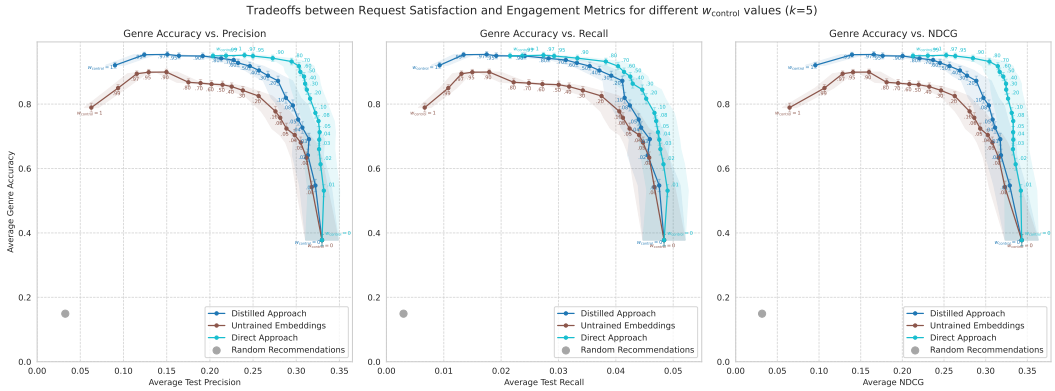

Figure 7: Performance comparison for genre-specific requests with k=5.

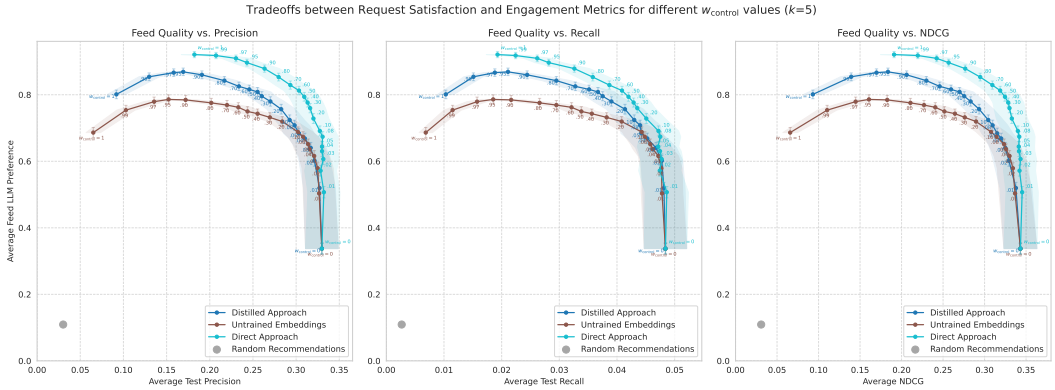

Figure 8: Performance comparison for open-ended requests with k=5.

Figure 8 shows that our approach can significantly improve request satisfaction while maintaining strong engagement metrics. The y-axis shows feed-level LLM preference scores, which evaluate how well the entire set of recommendations aligns with the user's request. As with genre-specific requests, we observe relatively favorable trade-offs: substantial improvements in request satisfaction can be achieved with modest reductions in engagement metrics.[6]

Our trade-off curves are likely conservative. Our quantitative results likely underestimate the practical impact of our method. The trade-off curves are based on held-out engagement data that reflects users' natural behavior patterns, completely independent of any requests. When we increase w_{control} to better satisfy user requests, the engagement metrics drop because we're diverging from these historical patterns - but this drop may be artificial.[7] In practice, when users explicitly request certain content, they are likely to engage with recommendations that satisfy that request, even if it differs from their typical engagement patterns. In light of this, the qualitative behavior of our method is likely to be more informative of the performance of our method.

[6]To be able to interpret what this increase in LLM feed-level score actually means, we need to be able to interpret the score. For this purpose, we show below some examples of feeds corresponding to specific user requests, and their respective scores.

[7]We attempt to minimize this issue by generating requests that are consistent with users' historical preferences (see Section 3.1). However, some mismatch between simulated requests and natural test set behavior is inevitable.

Impact of Recommendation List Length. Figure 9 shows the same analysis as Figures 7 and 8 in Section 3.2, but evaluating the top-10 recommendations rather than top-5. The results are similar, suggesting that our method's ability to balance engagement and request satisfaction is robust to the length of the recommendation list.

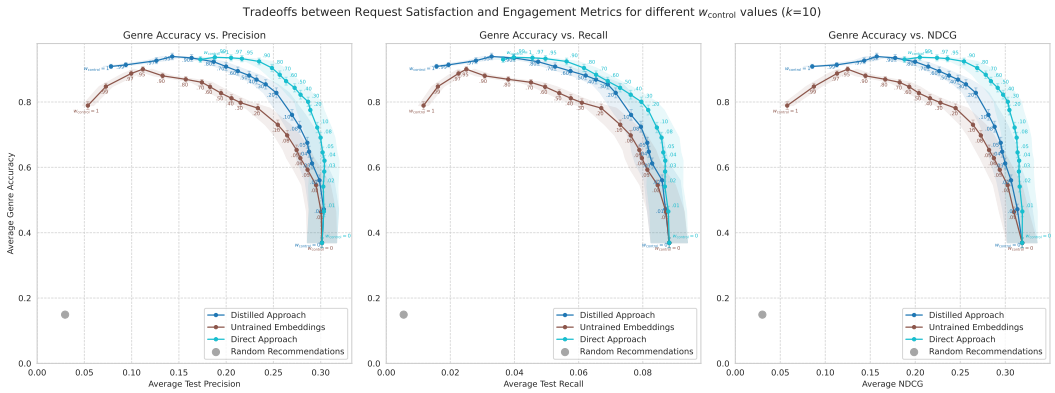

Figure 9: Performance comparison for genre-specific requests with k=10.

Impact of User History Context. An interesting question is whether providing the LLM with additional context about the user's engagement history improves its ability to judge how well items match the user's request. While intuitively, knowing that a user has previously engaged with certain items might help interpret their current request more accurately, our experiments so far suggest this additional context has minimal impact. Figures 10 and 11 compare performance when the LLM is given access to the user's 20 most recent interactions versus no historical context. The nearly identical performance curves suggest that the explicit request alone provides sufficient context for the LLM to make meaningful judgments about item-request alignment.

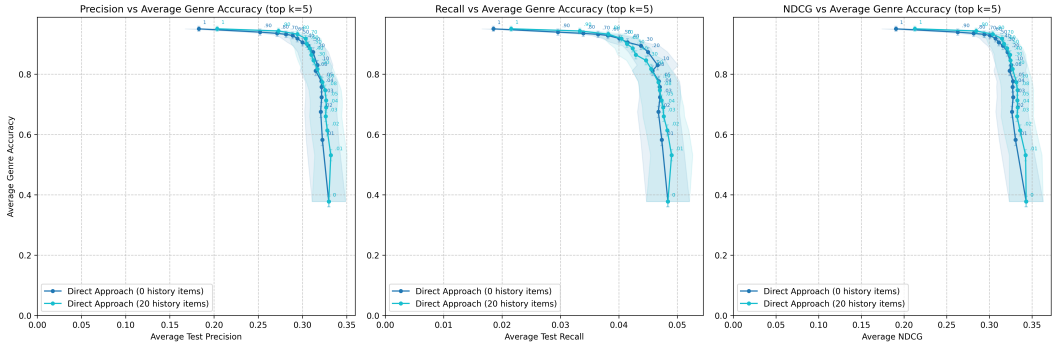

Figure 10: Comparison of performance with and without user history context (k=5). The blue line shows results when the LLM has access to up to 20 previous user interactions, while the orange line shows results without this context.

C.3 Additional Qualitative Results

The tables that follow have additional qualitative results.

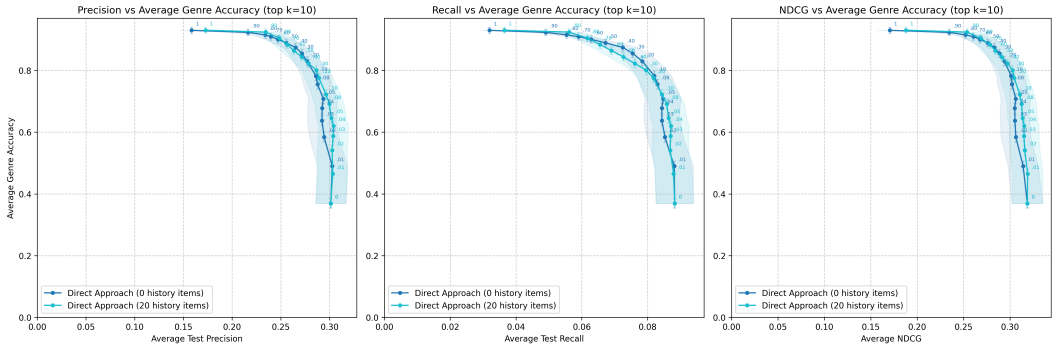

Figure 11: Same comparison as Figure 10 but with k=10, showing that the (lack of) impact of user history context is consistent across different recommendation list lengths.

Table 3: Impact of w_{control} on recommendations when user requests "Can you give me a good movie for a fun night with my kids?". As w_{control} increases from 0 to 1, recommendations shift from high-engagement mainstream films to children's content, demonstrating the trade-off between engagement and request satisfaction.

Title	**Genres**	**Request-aware Score**	**Eng. Value**	**Combined**
$w_{control} = 0$*: Pure Engagement-Based (No Request Consideration)*				
The English Patient	Drama, Romance, War	0.474	1.000	1.000
Air Force One	Action, Thriller	0.641	0.998	0.998
Contact	Drama, Sci-Fi	0.653	0.996	0.996
The Full Monty	Comedy	0.664	0.954	0.954
Titanic	Action, Drama, Romance	0.599	0.943	0.943
$w_{control} = 0.8$*: In-between weighting*				
The Full Monty	Comedy	0.664	0.954	0.722
Contact	Drama, Sci-Fi	0.653	0.996	0.722
E.T.	Children, Drama, Fantasy, Sci-Fi	0.897	2.16e-5	0.718
Winnie the Pooh	Animation, Children	0.896	5.39e-7	0.717
Air Force One	Action, Thriller	0.641	0.998	0.712
$w_{control} = 1$*: Pure Request-Based (No Engagement Consideration)*				
E.T.	Children, Drama, Fantasy, Sci-Fi	0.897	2.16e-5	0.897
Winnie the Pooh	Animation, Children	0.896	5.39e-7	0.896
Homeward Bound II	Adventure, Children	0.824	1.10e-8	0.824
Zeus and Roxanne	Children	0.821	9.08e-9	0.821
A Kid in King Arthur's Court	Adventure, Children, Comedy	0.812	1.51e-8	0.812

Table 4: Impact of w_{control} on recommendation rankings when user requests "Can you give me a good movie for a fun night with my 8 year old kids?". This is the same as Table 2 but using the direct LLM scoring method instead of the distilled method.

Title	Genres	Request Rank	Eng. Rank	Combined Rank
$w_{control} = 0$*: Pure Engagement-Based (No Request Consideration)*				
Pulp Fiction	Crime, Drama	1486	1	1
Army of Darkness	Action, Adventure, Comedy, Horror, Sci-Fi	211	2	2
GoldenEye	Action, Adventure, Thriller	700	3	3
The Fifth Element	Action, Sci-Fi	324	4	4
Desperado	Action, Romance, Thriller	703	5	5
$w_{control} = 0.5$*: Equal weighting*				
Dumbo	Animation, Children, Musical	12	23	18
Toy Story	Animation, Children, Comedy	3	41	22
The Lion King	Animation, Children, Musical	45	10	28
The Nightmare Before Christmas	Children, Comedy, Musical	48	8	28
Beauty and the Beast	Animation, Children, Musical	53	6	30
$w_{control} = 1$*: Pure Request-Based (No Engagement Consideration)*				
Winnie the Pooh	Animation, Children	1	258	1
E.T.	Children, Drama, Fantasy, Sci-Fi	2	135	2
Toy Story	Animation, Children, Comedy	3	41	3
Monty Python's Life of Brian	Comedy	4	74	4
Home Alone	Children, Comedy	5	247	5